LPI-Japan主催

HTML5プロフェッショナル認定試験 レベル1　ポイント解説無料セミナー

2015年10月

講師: 鈴木雅貴

(NTTソフトウェア株式会社)

自己紹介

- 鈴木雅貴(すずきまさたか)
  - NTTソフトウェア株式会社
    - HTML5アカデミック認定校
  - HTML5推進室所属
  - HTML5アカデミック認定校 セミナー講師
  - レベル1はβ試験にて認定取得
- 個人で日本語組版周り仕様の翻訳(停滞中)
  - http://suzukima.github.io/css-ja/

本日お話すること

- 試験について
  - 試験概要・範囲
- 試験範囲のポイント解説
  - Webの基礎知識
  - 要素
  - CSS3
  - レスポンシブWebデザイン
  - オフラインWebアプリケーション
- 学習の進め方

試験について

試験概要

- 公式サイトに情報があります
  - http://html5exam.jp/outline/

- JavaScriptは出ません
  - ただ、JavaScriptが何なのかや、使い方などは範囲に含まれるので、知っておいた方がよいでしょう

試験範囲

- 公式サイトに情報があります、が…
  - http://html5exam.jp/outline/objectives_lv1.html

6

試験範囲は広い

- 時間も限られていますので、本日は広い試験範囲を学習する上でのとっかかりとなる情報とポイントを持って帰ってもらうことを目的とします

- (X)HTML4.01/CSS2.1までも試験範囲には含まれますが、ここでの解説は一部を除き省略させていただきます

7

0. HTML5とは

HTML5とは

- HTML4.01までは、静的なWebページを作成するためのものであった
- しかし、Webの世界は、リッチなマルチメディアが多種多様なデバイスで実行されるアプリケーションのプラットフォームとなることが求められていた
- HTML5はそれを実現するために策定された仕様

9

HTML5 狭義と広義

- 狭義のHTML5は、W3Cが勧告したHTML5仕様のみ
  - A vocabulary and associated APIs for HTML and XHTML
    http://www.w3.org/TR/html5/

- 広義のHTML5は、狭義HTML5仕様にとどまらず、関連するCSSや各種JavaScript API群を含んだもの
  - あまりにも広大なため、W3Cでは技術分野を8つに分類して整理
    - セマンティクス/マルチメディア/オフライン&ストレージ/3D,グラフィックス,エフェクト/デバイスアクセス/パフォーマンス&インテグレーション/コネクティビティ/CSS3

10

をふまえて

1. Webの基礎知識

Webの基礎知識について

- http://html5exam.jp/outline/objectives_lv1.html#lv1_11
- 一見HTML5と関係が薄いように思えますが、HTML5によって、デザイナ/コーダ・Webプログラマ・サーバ/ネットワークエンジニアは、自分の担当部分だけでなく自らの領域を越えた知識やスキルが必要とされています

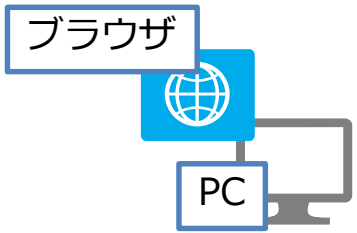
ポイント

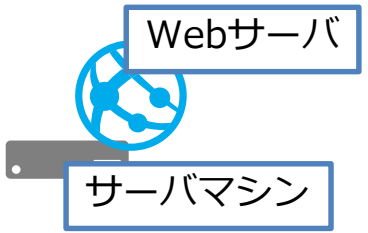
- Webサイトがどのような仕組みでWebブラウザに表示されるかを把握
  - IP, DNS, TCP, HTTP
- プロキシ2種

Webサイト
表示の仕組み

登場人物など

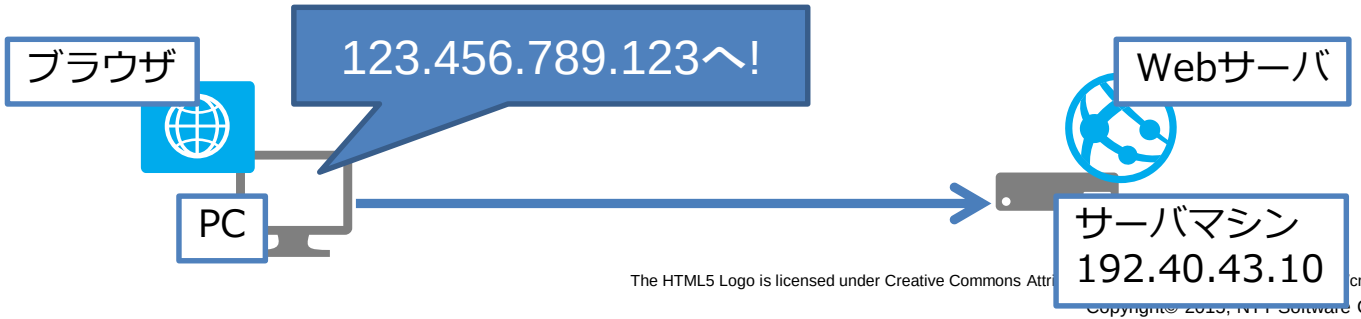
- PC: 略
- Webブラウザ: PCやスマホ上で動作する、Webサイト閲覧用ブラウザ(以降ブラウザ)
- サーバマシン: サーバアプリケーションが動作するコンピュータ機器
- Webサーバ: サーバマシン上でWebサイトを提供するサーバソフトウェア
- ざっくり概要で説明します





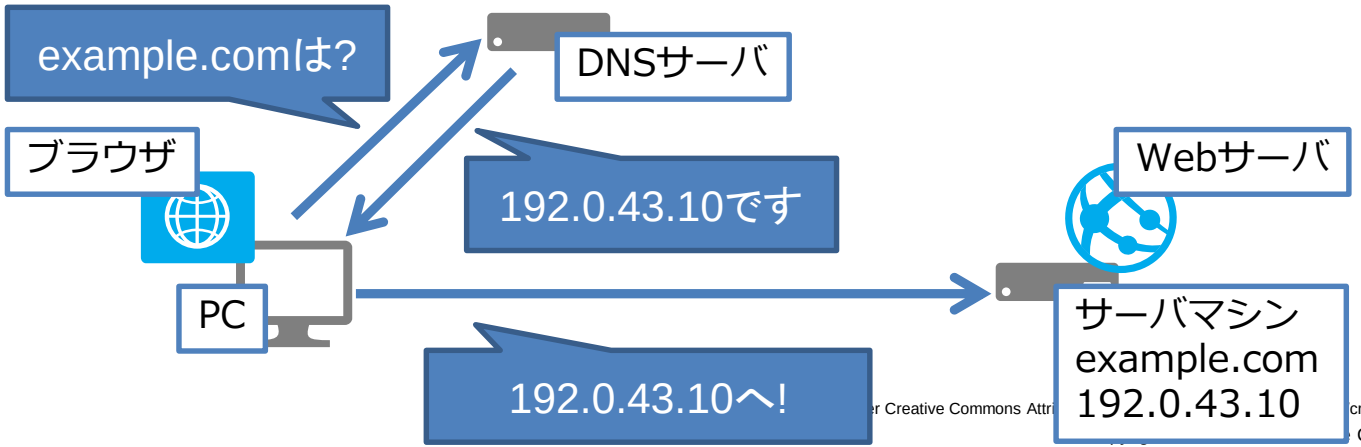
まず対象のWebサーバへアクセスしたい

- 対象のWebサーバをどう識別する?
  - ネットワーク上には無数のサーバが存在
- **IPアドレス**を使う
  - 192.0.43.10のような数字の羅列
  - IPアドレスを住所代わりにして目的のサーバにデータを送信
  - この仕組み(機器同士の通信)を規定しているのが **IP(Internet Protocol)**
  - IPは**情報を目的地まで届ける**ためのプロトコル



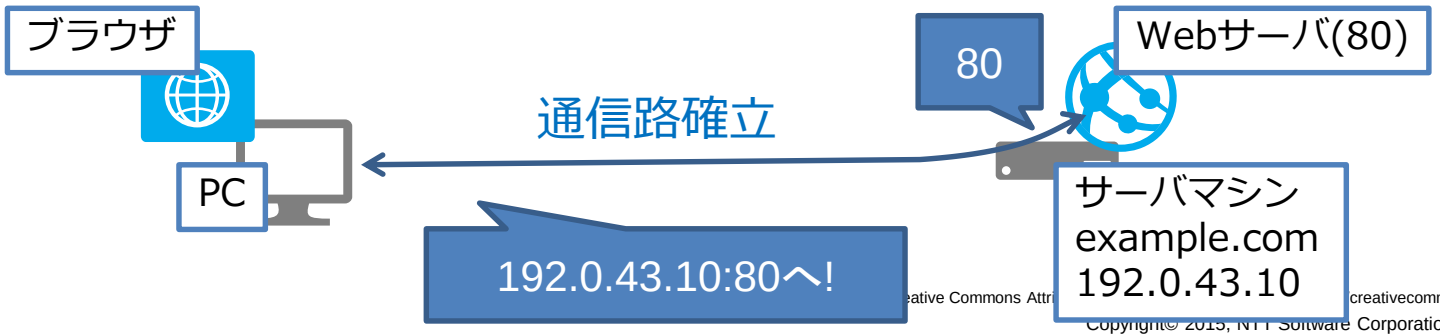
君はIPアドレスを覚えているのか？

- 普段IPアドレスを入力してませんよね
- DNS(Domain Name System)を使う
  - "example.com"のようなドメイン名と呼ばれる名前を付け、IPアドレスと紐づけて覚えやすくする(この仕組みがDNS)
  - DNSサーバに聞けば紐づいたIPアドレスがわかる(名前解決)
  - 詳しくはJPNICの「ドメイン名のしくみ」 https://www.nic.ad.jp/ja/dom/system.html



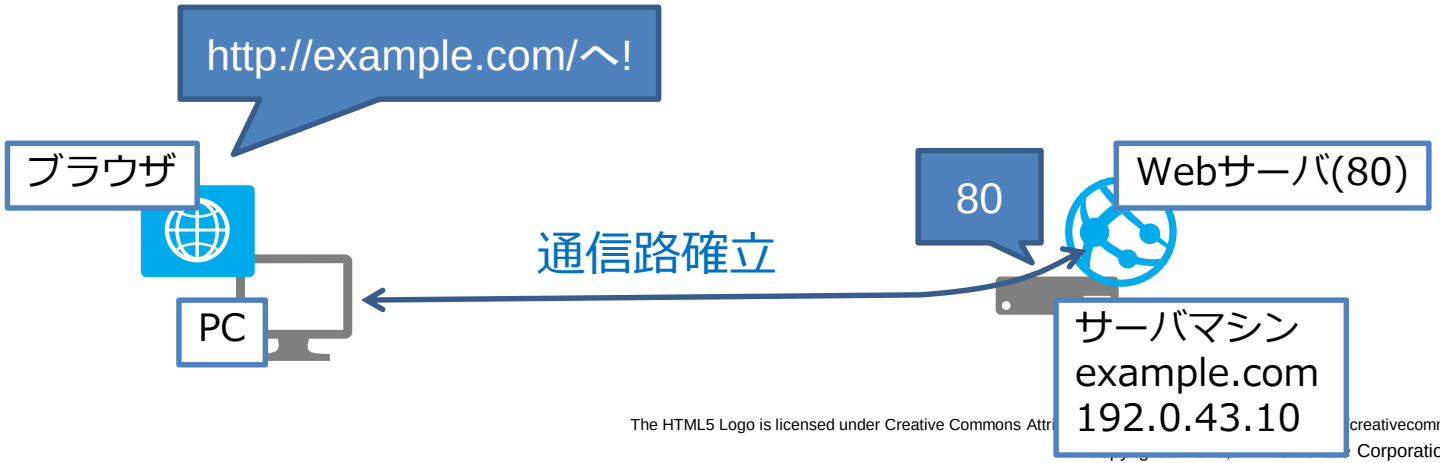
Webサーバはどれですか？

- Webサーバ以外にもサーバプログラムが
  - サーバマシンにはSSHなどのサーバプログラムも動作しており、IPアドレスだけではWebサーバを識別できない
- **ポート番号**を使う
  - サーバごとに既定のポート番号を指定(Webサーバは80番)
  - このあたりのアプリケーション間通信の仕組みを規定しているのが**TCP(Transmission Control Protocol)**
  - TCPはIPを利用し**通信路を確立**、そのうえで情報をやりとりする
  - TCPにはエラーチェックや再送など、**情報を確実に送る**仕組みがある



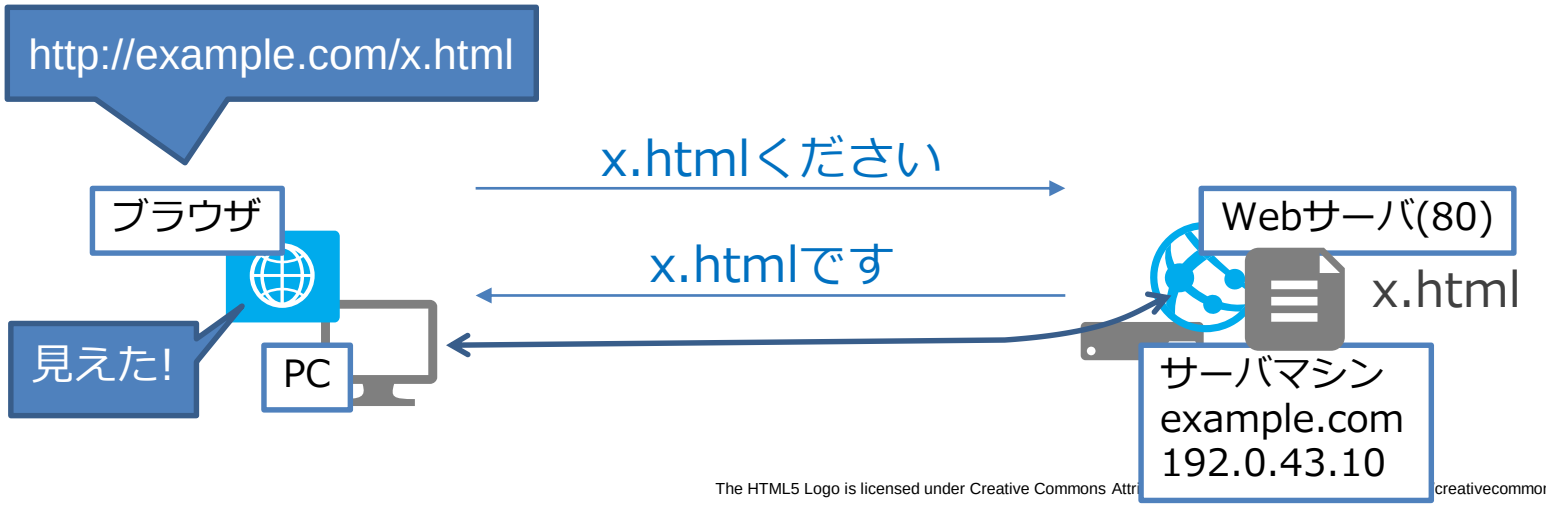
ポート番号、指定してますか？

- 普段はポート番号を入力しないですよね
- URLで通信プロトコル(ポート番号)を指定している
  - "http://example.com/"のhttp://で、HTTP(HyperText Transfer Protocol)を使っての通信を宣言している
  - HTTPは80番ポートを使用する(前述のWebサーバは80番と同意)、80番ポートを指定したことになる
- これでブラウザとWebサーバが無事つながった



つながったのでコンテンツのデータを取得

- HTTPでコンテンツを取得する
  1. 確立されたTCP通信路上で、ブラウザからWebサーバへ、x.htmlをくださいというリクエストを送信する
  2. Webサーバはブラウザへレスポンスとしてx.htmlの中身を返す
  3. TCPで確立した通信路を閉じる
  4. ブラウザはx.htmlを表示する



HTTPメッセージのポイント(1)

- HTTPメッセージはヘッダとボディに分かれる
  - ヘッダはリクエストやレスポンスがどのようなものかを示す情報が格納されている
    - クライアントやサーバの処理に必要な重要情報
  - ボディはデータ送信時にデータを格納する箇所
- リクエストで使用可能なメソッド(ヘッダに書かれる)

メソッド名	説明
GET	サーバから指定URIのリソースを取得
POST	クライアントからサーバへデータを送信 送信後サーバからデータが送られることもある
PUT	サーバの指定URIにデータを保存
HEAD	GETと同様だがヘッダのみ取得
OPTIONS	指定URIがサポートするメソッドを取得
DELETE	指定URIのリソース削除
TRACE	サーバまでのネットワーク経路チェック
CONNECT	TCPトンネル接続(プロキシ利用時のSSLトンネリングなどに使う)

HTTPメッセージのポイント(2)

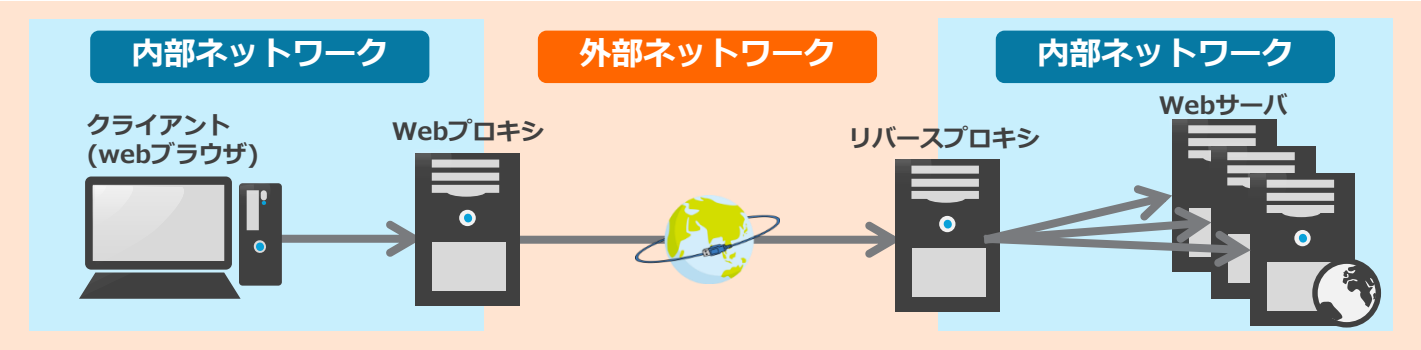
- レスポンスのステータスコード(ヘッダに書かれる)
  - リクエストの結果がどうだったかの情報が、3ケタの数字で書かれている

ステータスコード	概要と代表例
1xx	サーバ側の情報 100: Continue
2xx	成功 200: OK
3xx	転送 301: Moved Parmanently / 304: Not Modified
4xx	クライアントからのリクエストエラー 400: Bad Request / 401: Unauthorized / 403: Forbidden / 404: Not Found
5xx	サーバでのリクエスト処理エラー 500: Internal Server Error / 503: Service unavailable

プロキシ

プロキシ2種

- プロキシはWebブラウザ等クライアントとWebサーバとの通信を中継
- Webプロキシ(クライアント側に配置)
  - クライアントのインターネット直接続防止
  - キャッシュによる速度向上
- リバースプロキシ(Webサーバ側に配置)
  - Webサーバのインターネットからの直接続防止
  - Webサーバの負荷分散処理



2. 要素

要素について

- http://html5exam.jp/outline/objectives_lv1.html#lv1_13

- HTML5はもちろんのこと、HTML4.01以前も対象となっています

学習のポイント

- HTML4.01を押さえたうえで、5への変更点を学ぶ

- 何が変わったの？
  - 定型句がシンプルになっている
  - 要素に意味を持たせられるようになっている
  - スタイル的な機能は排除されている
  - 新しいインタフェースや機能が使えるようになっている

定型句がシンプルに

- **文書型宣言**

HTML4.01 Transitional

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

HTML5

```
<!DOCTYPE html>
```

- **文字エンコーディング**

HTML4.01 Transitional

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

HTML5(一例)

```
<meta charset="UTF-8">
```

- **他にもありますが、いろいろ楽になりました**

定型句がシンプルに

- たとえばこのような感じです

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>タイトル</title>
  <link rel="stylesheet" href="css/style.css">
  <script src="js/script.js"></script>
</head>
<body>
   …コンテンツ
</body>
</html>
```

要素に意味を持たせられるように

- セクションを明示する要素によりアウトラインを意識した作成が可能に
- セクションとは
  - 文書の一区分
  - アウトラインでは一階層を形成
- アウトラインとは
  - HTML文書では目次のようなもの
  - アウトラインのイメージ

1. あああ
2. いいい
    1. いいいのあああ
    2. いいいのいいい

セクションを明示する要素

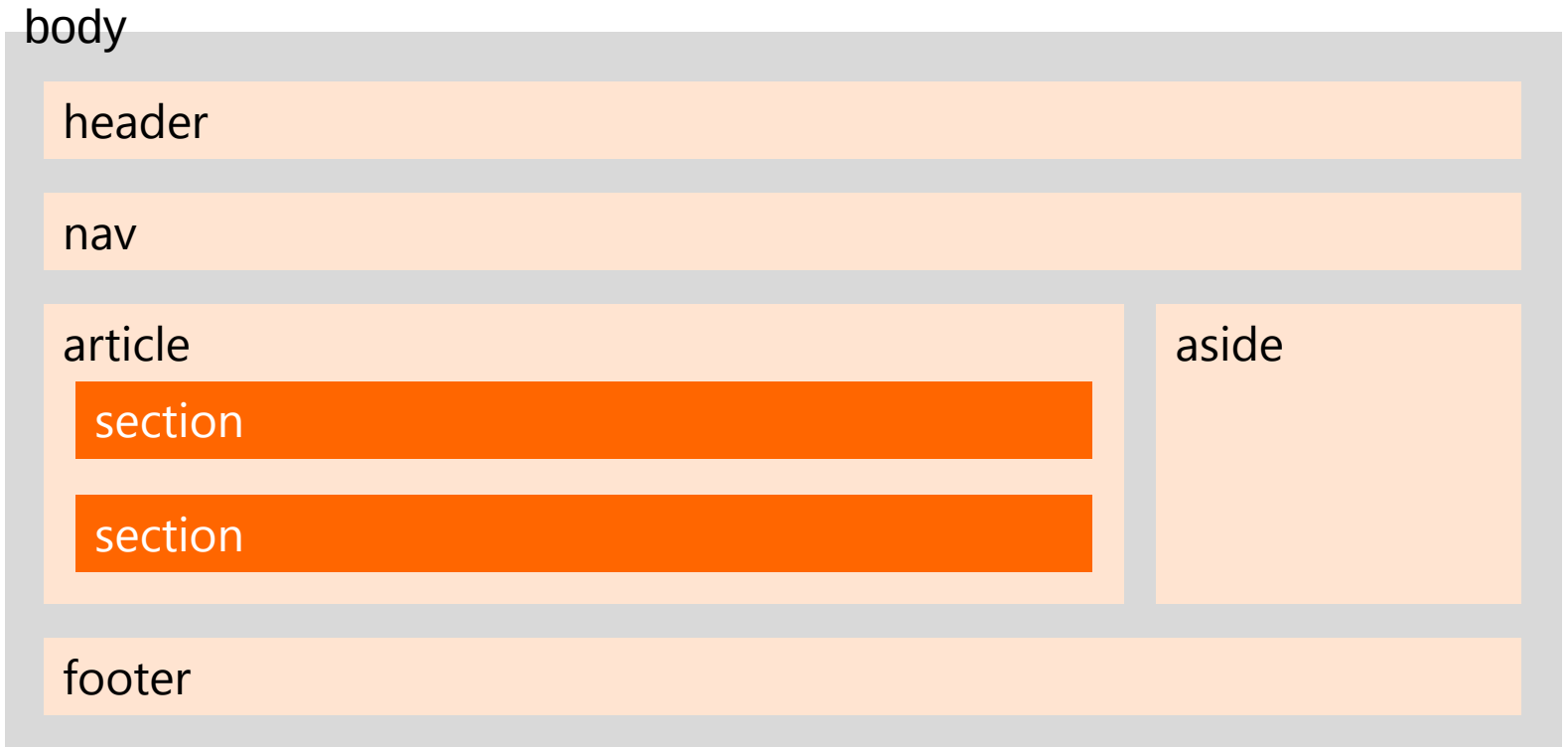
- 以下の要素でセクションが明示される

要素名	説明
section	一般的なセクションを表し、見出しとセットで使用
article	単体で完結できるセクション(1フィードに含める内容)
aside	あるコンテンツに関係しているが、切り離せるもの サイドバー、広告など
nav	主要なナビゲーション
body	body

- セクション明示はないが関連する要素

要素名	説明
h1~h6	セクションの見出し
header	セクションのヘッダ
footer	セクションのフッタ
address	直近祖先のセクションに関する連絡先

セクション関連要素の配置例



アウトライン生成の例

- 以下のHTMLがあったとする

```
<body>
  <h1>Apples</h1>
  <p>Apples are fruit.</p>
  <section>
    <h2>Taste</h2>
    <p>They taste lovely.</p>
      <section>
        <h3>Sweet</h3>
        <p>Red apples are sweeter than green ones.</p>
      </section>
  </section>
  <section>
    <h2>Color</h2>
    <p>Apples come in various colors.</p>
  </section>
</body>
```

- アウトラインを確認してみよう
  - validator.nu http://validator.nu/

意味を持たせられるメリット

- 構造がページの読解の助けとなる人々に大きな利点をもたらす
  - いわゆるアクセシビリティの向上
- これがオープンなプラットフォームとして提供されているのが大きなポイント
  - 誰でも使える！
- 構造はこうしなければダメ！という答えがあるわけではない
  - 構造には作成者の主張が反映される
  - 判断の参考になるものとして HTML5 DoctorのHTML5 Sectioning Element Flowchartがある

スタイル的な機能の排除

- **HTML５の要素からは、要素および属性によるスタイル指定機能は排除されている**
  - スタイル指定は基本CSSでやることに
    - 廃止された要素例: big, center, font, strike, tt
  - 排除された主要な理由は以下
    - アクセシビリティの低下
    - メンテナンスコストの増加
    - 文書サイズの増大
  - 残ったのはstyle要素および属性
- **スタイル的機能の要素の一部は機能を再定義されているのもポイント**
  - b, i, hr, s, small, u

新しいインタフェースや機能

- フォームのinput要素で使えるtype属性が大幅増加
  - http://www.w3.org/TR/html5/forms.html#states-of-the-type-attribute

- HTMLだけでマルチメディアコンテンツを再生できるaudio要素/video要素

- プラグインを使用せずビットマップ画像を描画できるcanvas要素

他のポイント

- HTML4.01からHTML5での変更点を押さえておく
  - 新規追加された要素
  - 変更となった要素・属性
  - 削除された要素・属性
- 各要素のカテゴリを把握しておく
  - http://www.w3.org/TR/html5/dom.html#kinds-of-content
  - http://www.w3.org/TR/html5/index.html#element-content-categories
  - できればコンテンツモデルも覚えておけるとよいですが、なかなか大変です
    - http://www.w3.org/TR/html5/index.html#elements-1

3. CSS3

CSS3について

- http://html5exam.jp/outline/objectives_lv1.html#lv1_12
- CSS3はもちろんのこと、CSS2.1以前も対象となっています

学習のポイント

- CSS2.1を押さえたうえで、3への変更点を学ぶ
- CSS2.1のポイント
  - カスケード
  - ボックスモデル
- CSS3で何が変わったの？
  - 画像が必要だった修飾がCSSで可能に
  - 便利セレクタの大幅追加
  - 簡単にマルチカラムレイアウト実現
  - CSSのみで変形、アニメーション実現

カスケード

カスケード

- カスケード処理
  - 継承などで1つの要素に対し複数の指定が起こりうるが、基本的に優先度は直接指定>継承(近い指定の方が強い)
  - それが同じ場合の優先度判断をするのがカスケード処理
- 優先順位づけルールその1: CSSの種類
  - 文章作成者CSS>ユーザCSS>ブラウザ標準CSS
    - ユーザCSSは、利用者がWebブラウザに対して指定するCSS
    - ブラウザ標準CSSは、Webブラウザの持つデフォルトのCSS
  - ただし、CSSの値に”!important”をつけると最優先される

例: pの文字サイズ指定に!importantを使用する

```
p { font-size: 1em !important; }
```

まとめると

!important付きユーザCSS > !important付き文書作成者CSS > 文章作成者指定CSS > ユーザ指定CSS > ブラウザ標準CSS

カスケード

- **優先順位づけルールその2: セレクタタイプ**
  - **CSSの種類が同じである場合、セレクタをタイプ別に分類し、それぞれの個数を数え、点数(詳細度)を算出し、高い方を優先する**
- **セレクタのタイプ**

タイプ	対象
A	style属性に記述したCSS
B	IDセレクタ
C	クラスセレクタ、属性セレクタ、疑似クラス
D	要素セレクタ、疑似要素

- **詳細度の算出**

  詳細度は個数を連結した”A.B.C.D”と書くとわかりやすい
  Aが0つ、Bが1つ、Cが2つ、Dが1つの場合の詳細度は”0.1.2.1”
  比較方法は後述

カスケード

- **詳細度からの優先度算出**
  - **A>B>C>Dの順で優先度が高い**
  - **優先度が高いセレクタタイプの点数が高い方が優先される**

詳細度からの優先度算出の例

```
#id { 装飾A }          →0.1.0.0
ul li.class { 装飾B }  →0.0.1.2
ui ol+li { 装飾C }     →0.0.0.3
```

この場合の優先度順は装飾A>装飾B>装飾Cとなり、装飾Aが適用される

Specify Calculatorは詳細度を算出してくれるので参考に

カスケード

- **優先順位づけルールその3: 出現順**
  - **詳細度による優先度も同じ場合、後に出現したCSSが優先される**

出現順からの優先度の例

```
#id { 装飾A }
#id { 装飾B }
```

この場合は後に出現した装飾Bが適用される

ボックスモデル

ボックスモデルとボックスサイズ算出

- 文書内の全要素は四角形の領域を形成
  - コンテンツ領域、padding(ボックス内側の余白)、border(境界線)、margin(ボックス外側の余白)の4つから成り立つ

- ボックスサイズの算出
  - 幅ならコンテンツ領域の幅(width)+padding+border
  - プロパティbox-sizingの値をborder-boxにすると、ボックスサイズがwidthもしくはheightで指定した値となる
    - コンテンツ領域のサイズはpaddingやborderの値も使って自動算出

CSS3変更点

CSS3で可能になった修飾の例

- ボックスの角丸
  - border-radiusプロパティ(http://www.w3.org/TR/css3-background/#corners)
- グラデーション
  - linear-gradient, radial-gradient(http://www.w3.org/TR/css3-images/#gradients)
- ボックスに影
  - box-shadowプロパティ(http://www.w3.org/TR/css3-background/#the-box-shadow)

便利セレクタの大幅追加(以下は一部)

- 属性セレクタに前方/後方/部分一致が追加
  - [attr^="val"], [attr$="val"], [attr*="val"]
  - http://www.w3.org/TR/css3-selectors/#attribute-substrings
- 結合子に兄弟セレクタ"~"が追加
  - http://www.w3.org/TR/css3-selectors/#attribute-substrings
- 構造を利用した疑似クラスが大幅追加
  - nth-child(), nth-of-type(), last-childなど
  - http://www.w3.org/TR/css3-selectors/#general-sibling-combinators

簡単にマルチカラムレイアウト

- Multi-column Layout Moduleで、複数段組みレイアウトを簡単に実現可能
  - column-count, coloum-gapプロパティなど
  - http://www.w3.org/TR/css3-multicol/
- Flexible Box Layout Moduleで、floatを使わずにボックスを横並びにできる
  - display: flexとflex-directionプロパティなど
  - http://www.w3.org/TR/css-flexbox-1/
- CSSによるレイアウトの自由度が増しています

CSSで変形、アニメーション

- transformプロパティでボックスを移動・変形できる
  - translate(), rotate(), scale(), skew()など
  - http://www.w3.org/TR/css3-transforms/
- transition関連プロパティで、要素の変化を滑らかにできる
  - 例えば、:hoverでの変化を滑らかにするなど
  - http://www.w3.org/TR/css3-transitions/
- animation関連プロパティで、要素の変化を連続して行える
  @keyframesを指定するのが特徴
  - http://www.w3.org/TR/css3-animations/

4. レスポンシブWebデザイン

レスポンシブWebデザインについて

- http://html5exam.jp/outline/objectives_lv1.html#lv1_14

- タイトルはレスポンシブWebデザインですが、スマートデバイスに特化した内容も盛り込まれているので要注意

学習のポイント

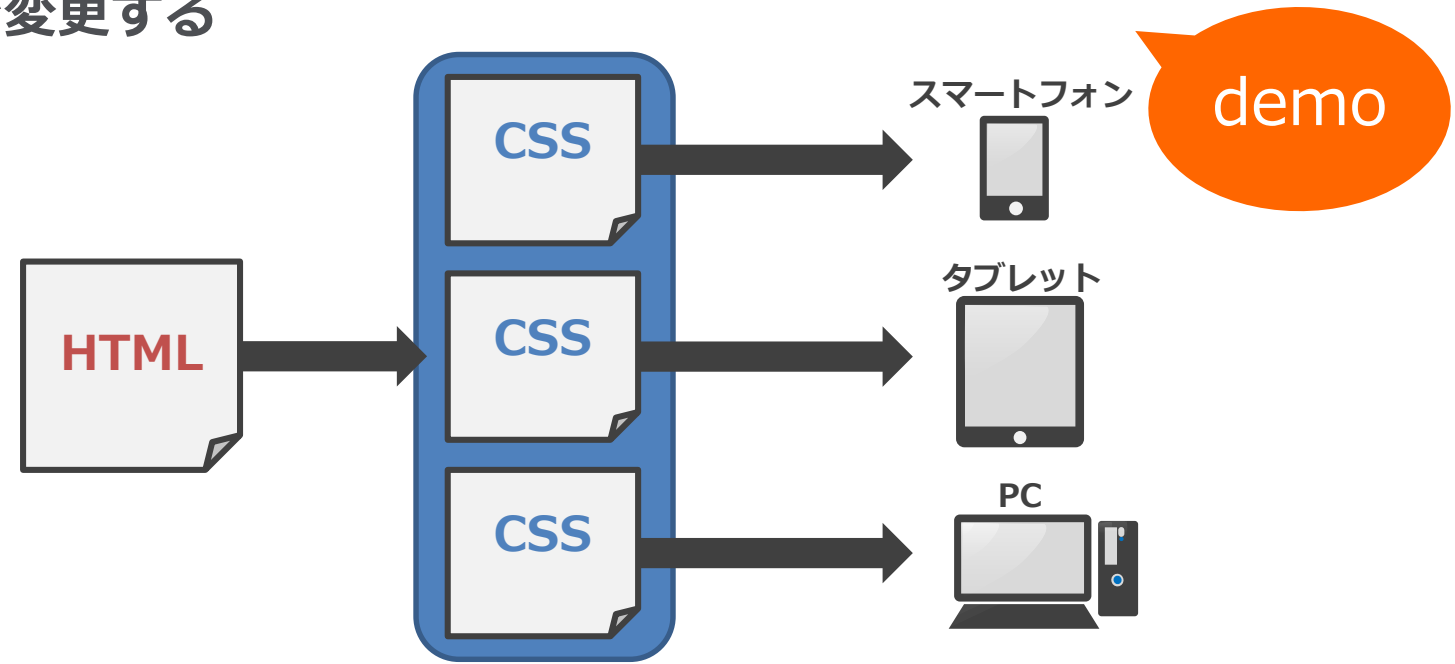
- レスポンシブWebデザインがどういうものかを押さえる
- 実現方法を押さえる

レスポンシブWebデザインとは

- 多種多様なデバイスが登場し続ける状況の中、デバイスごとに対応サイトを開発するのはコストがかかり、また将来登場する端末への対応も不明
- そこで、レスポンシブWebデザインと呼ばれるWebサイト開発手法が登場
  - 広い意味では利用中であるユーザの状況や環境に合わせ、そのユーザにより良い体験を提供しようという考え方
  - 手法を表す場合には、1つのHTMLで、デバイスの特性(主に画面横幅)に応じてレイアウトやデザインを変更する手法を指す

デバイス特性に応じてレイアウトを変更する方法

- レスポンシブWebデザインでは、デバイス特性を利用してCSSを切り替えるMedia Queriesを利用し、デバイスごとにレイアウトを変更する



メリットとデメリット

- メリット
  - 将来登場するデバイスも見越して対応可能
  - URLが同じとなるため、SEO的に有利
  - リダイレクトが発生しないため読み込み時間を短縮可能
  - HTMLが１つで済むのでコストを削減できる(可能性)
- デメリット
  - 設計・製造の難易度が高い
    - 逆にコスト高になることも
  - モバイル向けには動画や画像も切り替える必要
    - モバイルは回線が不安定で画面は小さい
  - 画面フローの変更には対応できない
  - スマホ用にPC向けサイトを表示できない
  - HTML/CSSのサイズが増加し、ページ表示や動作が重くなる可能性

レスポンシブWebデザインの実現手法

- Media Queries(前述)
- Fluid Grid
  - 画面サイズにあわせてグリッドの幅を変更する
- Fluid Image
  - 画面サイズに合わせて画像サイズを変更する

- これ以外にもあるが基本はこの3つ
- フレームワークをベースに作ることも多い

5. オフラインWeb アプリケーション (概要とマニフェスト)

オフラインWebアプリケーションについて

- http://html5exam.jp/outline/objectives_lv1.html#lv1_15

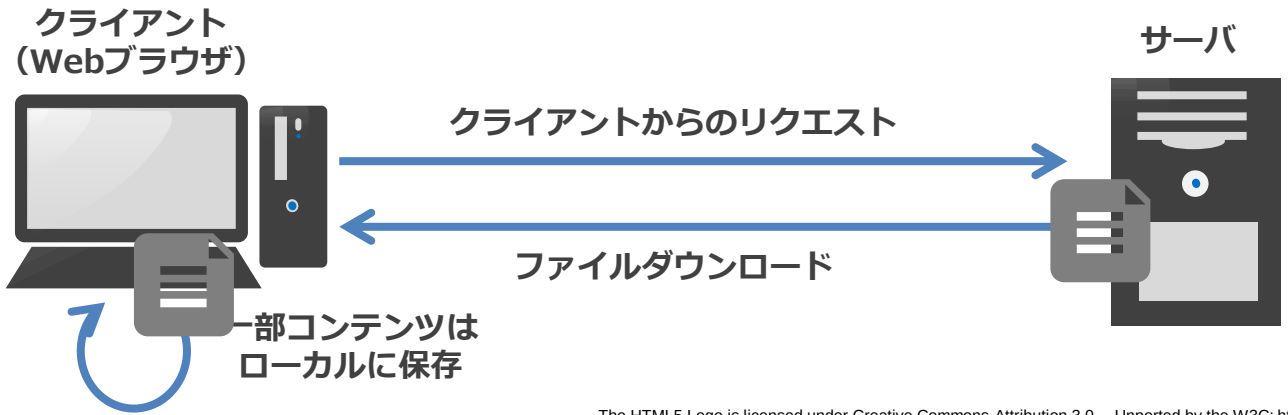
学習のポイント

- 概要と
- マニフェスト

オフラインWebアプリケーションとは

- ブラウザがオフラインでも静的コンテンツを閲覧可能なWebアプリケーション

- ブラウザは、キャッシュマニフェストというファイルを読み取り、キャッシュマニフェストに指定されたコンテンツをローカルに保存
  ⇒この仕組みによりオフライン閲覧が可能



オフラインWebアプリケーション利用時の動作

- 次回からは、ローカルへ保存したコンテンツを閲覧
  - オンラインでなくてもコンテンツを閲覧できる
  - オンラインであってもサーバアクセスが不要

メリットとデメリット

- メリット
  - オフライン状態でもWebアプリケーションを閲覧させることが可能
  - (2回目のアクセス以降)ローカルファイルはネットワーク経由のファイルより速く読み込まれる
  - 全ファイルをサーバから取得しないため、サーバ負荷の軽減が可能
- デメリット
  - サーバ側のファイルを更新した場合、ユーザのローカルキャッシュファイルも更新が必要
  - キャッシュマニフェストの記述を誤った場合、変更ファイルが更新されないことがある

実現のために必要な作業

- Webサーバの設定
  - ".manifest"のMIMEタイプを"text/cache-manifest"に
  - これがないとキャッシュマニフェストファイルが認識されない
- キャッシュマニフェストファイルの作成
  - どのファイルをキャッシュする/しないをなどを設定する
  - 書き方は後述
- キャッシュ対象HTMLファイルでのmanifest属性指定
  - html要素のmanifest属性に利用するキャッシュマニフェストファイルのパスを指定
  - HTMLファイルのみでよい(CSS/JavaScript等は不要)

Apacheや.htaccessの設定

ファイルを作成してWebサーバに配置

HTMLに追記

キャッシュマニフェストファイルの記述方法

```
CACHE MANIFEST
# 先頭文字を#で始めることにより、コメントを記述可能
# 本例ではパスをキャッシュマニフェストからの相対パスで記載
# ここはCACHE:セクション
# このセクションにはローカルキャッシュするファイルを記載
index.html
app.js

NETWORK:
# このセクションにはキャッシュしないファイルを記載
submit.cgi

FALLBACK:
# このセクションには代替ファイルを記載
/ offline.html

SETTINGS:
# キャッシュの利用方法を設定可能
prefer-online
```

最後に: 学習の進め方

学習の進め方

- 実際に試してみよう
  - JSFiddleやJS Binならブラウザから書いてすぐ確認できる
  - 特にCSSは動かして確認することで理解が深まる
- 仕様書を見よう
  - CSSのプロパティなどはそうするのが確実
  - 全部を見る必要はありません
- 暗記も必要
  - 合格ラインは70%であることも考慮して効率的に
- 受験中にあせらない心構えを
  - 70%なので少々わからなくても平気

役に立つ情報源など

- Mozilla Developers Network（MDN）の Web technology for developers
  - https://developer.mozilla.org/ja/docs/Web
  - ここからHTMLやCSSの技術情報が見られる
  - かなり日本語化されています

- W3C CSSプロパティ一覧
  - http://www.w3.org/Style/CSS/all-properties
  - プロパティに関連する仕様とその標準化進行状況がわかる

- 資格関連情報
  - 公式サイト http://www.html5exam.jp/
  - Twitter @HTML5Cert
  - Facebook https://www.facebook.com/html5exam

71

ふりかえり: 本日お話したこと

- 試験について
  - 試験概要
  - 試験範囲

- 試験範囲のポイント解説
  - Webの基礎知識
  - 要素
  - CSS3
  - レスポンシブWebデザイン
  - オフラインWebアプリケーション

- 学習の進め方

72

おまけ

ベンダプレフィックス

- CSSで標準化の進む機能がブラウザで先行実装された場合、プロパティや値の先頭にベンダプレフィックスと呼ばれる特定の接頭語を付ける必要がある
- 例えば、Firefoxのベンダプレフィックスは以下

```
-moz-プロパティA: 値;
```

  - どのようなベンダプレフィックスがあるかはベンダ固有の話なので、試験にはでないと思われます
- 現状は次のようになっている
  - MozillaやGoogleは今後ベンダプレフィックスをつけないと表明
  - ベンダプレフィックスは標準化がある程度進むと外すことが推奨
  - ただ昔のブラウザを対象にすることはある
- 現状はベンダプレフィックスあり/なしを併記しておくのがよいでしょう

ブラウザでの対応状況調査方法

- 各ブラウザにおける要素やCSS機能の対応状況、ベンダプレフィックスの有無などは頻繁に変更される
  - 自分で調べているとつらい
- caniuseで調査するとよい
  - http://caniuse.com/

- 使い方は検索フォームで使いたい機能を検索するか、一覧から選
- ぱっと調べるには非常に便利です
- 試験には出ません

LPI-JAPAN HTML5 Professional Certification

Open the Future with HTML5.